



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/052,242	01/23/2002	Tuomo Aro	P 290636 2011615US/A/kp	8023

909 7590 01/10/2005
PILLSBURY WINTHROP, LLP
P.O. BOX 10500
MCLEAN, VA 22102

EXAMINER

MITCHELL, JASON D

ART UNIT PAPER NUMBER

2124

DATE MAILED: 01/10/2005

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/052,242	ARO ET AL.	
	Examiner	Art Unit	
	Jason Mitchell	2124	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --
Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 23 January 2002.
- 2a) ☐ This action is **FINAL**. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-32 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-32 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☒ The specification is objected to by the Examiner.
- 10) ☒ The drawing(s) filed on 23 January 2002 is/are: a) ☐ accepted or b) ☒ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
 2. ☐ Certified copies of the priority documents have been received in Application No. _____.
 3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- | | |
|--|---|
| 1) <input checked="" type="checkbox"/> Notice of References Cited (PTO-892) | 4) <input type="checkbox"/> Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____ |
| 2) <input type="checkbox"/> Notice of Draftsperson's Patent Drawing Review (PTO-948) | 5) <input type="checkbox"/> Notice of Informal Patent Application (PTO-152) |
| 3) <input checked="" type="checkbox"/> Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date <u>1/23/02</u> | 6) <input type="checkbox"/> Other: _____ |

DETAILED ACTION

1. This action is in response to an application filed on 1/23/2002.
2. Claims 1-32 are pending in this case.

Drawings

3. **The drawings are objected to as failing to comply with 37 CFR 1.84(p)(4) because reference characters “301-314” have been used to designate objects in both Figs. 3 A and B and Figs. 4 A and B. Corrected drawing sheets in compliance with 37 CFR 1.121(d) are required in reply to the Office action to avoid abandonment of the application.**
4. **The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they include the following reference character(s) not mentioned in the description: 308, 313 and 314 from Fig. 4A and 701 and 811 from Figs 7 and 8, respectively. Corrected drawing sheets in compliance with 37 CFR 1.121(d), or amendment to the specification to add the reference character(s) in the description in compliance with 37 CFR 1.121(b) are required in reply to the Office action to avoid abandonment of the application.**
5. **The drawings are objected to as failing to comply with 37 CFR 1.84(p)(5) because they do not include the following reference sign(s) mentioned in the description: step 380 and serialization layer 500 referenced in paragraphs [0051] and [0055] respectively. Corrected drawing sheets in compliance with 37 CFR**

1.121(d) are required in reply to the Office action to avoid abandonment of the application.

6. Any amended replacement drawing sheet should include all of the figures appearing on the immediate prior version of the sheet, even if only one figure is being amended. The replacement sheet(s) should be labeled "Replacement Sheet" in the page header (as per 37 CFR 1.84(c)) so as not to obstruct any portion of the drawing figures. If the changes are not accepted by the examiner, the applicant will be notified and informed of any required corrective action in the next Office action. The objection to the drawings will not be held in abeyance.

Claim Objections

Claim 19 is objected to because of the following informalities: Claim 19 fails to further limit the parent claim (18). 'a distributed computer system' as described in claim 18 necessarily includes a server – client relationship as recited in claim 19. Appropriate correction is required.

7. **Claim 32 is objected to under 37 CFR 1.75(c) as being in improper form because a multiple dependent claim cannot depend from any other multiple dependent claim. See MPEP § 608.01(n).** Accordingly, the claim has not been further treated on the merits.

Claim Rejections - 35 USC § 112

8. The following is a quotation of the second paragraph of 35 U.S.C. 112:

Art Unit: 2124

The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

9. **Claim 2 is rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.** The claim recites 'delivering each of said dedicated data exchange metadata descriptions' both 'with the delivery of the respective version of said second software component', and 'separately during each data exchange session'. One of ordinary skill in the art would not be able to determine definitively how, when and where the data exchange metadata descriptions should be delivered. For the sake of this examination, Examiner's best understanding will be used and the claim will be taken to mean delivering the dedicated data exchange metadata descriptions with the delivery of the second software component and later passing the dedicated data exchange metadata descriptions, or a reference to them, to the new version of said software component during a handshake procedure.

10. **Claim 1 recites the limitation "said metadata description" in line 12. There is insufficient antecedent basis for this limitation in the claim.**

11. **Claim 26 recites the limitation "said new version" in line 7. There is insufficient antecedent basis for this limitation in the claim.**

12. Applicant is advised that several other similar 'antecedent basis' problems exist, through out the claims. Applicant's cooperation is requested in correcting any such errors.

Claim Rejections - 35 USC § 102

13. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(e) the invention was described in (1) an application for patent, published under section 122(b), by another filed in the United States before the invention by the applicant for patent or (2) a patent granted on an application for patent by another filed in the United States before the invention by the applicant for patent, except that an international application filed under the treaty defined in section 351(a) shall have the effects for purposes of this subsection of an application filed in the United States only if the international application designated the United States and was published under Article 21(2) of such treaty in the English language.

14. Claims 1-2, 4-5 and 18-24 are rejected under 35 U.S.C. 102(e) as being anticipated by US 6,658,625 to Allen.

Regarding Claim 1: Allen discloses a method for updating a software component in a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and one or more second software components (Fig. 1, Application 123), said method comprising: updating said first software component with a new version of software (col. 6, lines 42-44 'multiple versions of server programs') substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components (col. 3, lines 7-10 'interprets a data description ... that can accommodate changes in the data'); providing said updated software version with a data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD')

Allen does not expressly disclose delivering said metadata description with said new version of said software component, he does disclose creating new versions of the data descriptions when the server interface changes (col. 19, lines 44-46 'The data description can then be modified when necessary to encompass changes in the interface'), thereby inherently disclosing delivery.

Regarding Claim 2: The rejection of claim 1 is incorporated; further, Allen discloses providing a dedicated data exchange metadata description for each version of said second software component that should be compatible (Fig. 1, Data Description 124); each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230') and delivering said dedicated data exchange metadata description separately for each data exchange session during a session handshake procedure between said new version of said software component and the respective version of said second software component (col. 12, lines 2-4 'This data is then converted ... when application 123 requests the data').

Allen does not explicitly disclose delivering the data descriptions, but does show them residing on the client computer (Fig. 1, Data Description 124), thereby inherently disclosing the delivery of said dedicated data exchange metadata descriptions.

Regarding Claim 1/4: The rejection of claim 1 is incorporated; further Allen discloses said first software component installed on a server computer (col. 7, lines 51-52 'sever 185') to be used as a server software component (Fig. 1, Server Program 195) in a

client-server-type distributed software system (col. 7, lines 51-52 'Client 100 communicates ... with sever 185'), said one or more second software components (Fig. 1, Application 123) being client software components located in respective client stations (col. 7, lines 28-30 'computer system 100 ... is a ...client computer system'). Allen does not explicitly disclose delivering the software components, but discloses the server software component (Fig. 1, Server Program 195) and the client software components (Fig. 1, Application 123) existing on server (Fig. 1, Server 185) and client computers (Fig. 1, Client 100), respectively, thereby inherently disclosing their delivery to the respective computers.

Regarding Claim 2/4: The rejection of claim 2 is incorporated; further Allen discloses said first software component installed on a server computer (col. 7, lines 51-52 'sever 185') to be used as a server software component (Fig. 1, Server Program 195) in a client-server-type distributed software system (col. 7, lines 51-52 'Client 100 communicates ... with sever 185'), said one or more second software components (Fig. 1, Application 123) being client software components located in respective client stations (col. 7, lines 28-30 'computer system 100 ... is a ...client computer system'). Allen does not explicitly disclose delivering the software components, but discloses the server software component (Fig. 1, Server Program 195) and the client software components (Fig. 1, Application 123) existing on server (Fig. 1, Server 185) and client computers (Fig. 1, Client 100), respectively, thereby inherently disclosing their delivery to the respective computers.

Regarding Claim 5: Allen discloses a method for updating a software component in a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and one or more second software components (Fig. 1, Application 123), said method comprising: updating said first software component with a new version of software (col. 6, lines 42-44 'multiple versions of server programs') substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components (col. 3, lines 7-10 'interprets a data description ... that can accommodate changes in the data'); providing said updated software version with a data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD'); providing a dedicated data exchange metadata description for each version of said second software component that should be compatible (Fig. 1, Data Description 124); each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230').

Allen does not expressly disclose delivering said metadata description with said new version of said software component, he does disclose creating new versions of the data descriptions when the server interface changes (col. 19, lines 44-46 'The data

description can then be modified when necessary to encompass changes in the interface'), thereby inherently disclosing delivery.

Regarding Claim 18: Allen discloses a distributed computer system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and one or more second software components (Fig. 1, Application 123), exchanging serialized data with said first software component; at least one of said second software components being an older version substantially compatible with the functionality of said first software component but having incompatible data structures (col. 3, lines 7-10 'interprets a data description ... that can accommodate changes in the data') said first software component having a first data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD') by said first software component; said first software component having a dedicated second data exchange metadata description for at least one older version of said second software component (Fig. 1, Data Description 124) each said second data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230').

Regarding Claim 19: The rejection of claim 18 is incorporated; further Allen discloses said computer system is a client-server-type computer system (col. 7, lines 51-52 'Client 100 communicates ... with sever 185'), and wherein said first software component is a

server software component, (Fig. 1, Server Program 195) and wherein said second software component is a client software component (Fig. 1, Application 123).

Regarding Claim 18/20: The rejection of claim 18 is incorporated; further Allen discloses said first software component is installed in a first computer (Fig. 1, server 185), and wherein said one or more second software components are installed in one or more second computers (Fig. 1, client 100).

Regarding Claim 19/20: The rejection of claim 19 is incorporated; further Allen discloses said first software component is installed in a first computer (Fig. 1, server 185), and wherein said one or more second software components are installed in one or more second computers (Fig. 1, client 100).

Regarding Claim 21: Allen discloses A client-server computer system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a server software component (Fig. 1, Server Program 195) one or more client software components (Fig. 1, Application 123) exchanging serialized data with said server software component (col. 3, lines 16-20 'sent to and/or received from a server') said server software component having a first data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange by said server software component (col. 11, lines 42-44 'PCML DTD'), said server software component having a dedicated second data exchange metadata description for at least one older version of said client software component (Fig. 1, Data Description 124) each said second data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second

Art Unit: 2124

software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230').

Regarding Claim 22: The rejection of claim 21 is incorporated; further, Allen discloses creating a serialization scheme (col. 12, lines 17-20 'serializing the hash table ... to create PCML serialized') for said server software component and said older version of said client software component on the basis of said metadata descriptions of these versions (col. 11, lines 42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ... PCML Serialized 220 is an enhancement to the parser output') said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component (col. 11, line 66- col. 12, line 2 'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call server program 195 and to receive data ... from server program 195').

Regarding Claim 21/23: The rejection of claim 21 is incorporated; further Allen discloses a serialization routine for transforming data items into a stream of data bits to be sent to said client software component (col. 12, lines 20-27 'create a persistent object that is generally recorded to a file as a byte stream'), and a deserialization routine for transforming a stream of bits received from said client software components into data items (col. 12, lines 20-27 'file can be ... "rehydrated" back into its constituent objects').

Regarding Claim 22/23: The rejection of claim 22 is incorporated; further Allen discloses a serialization routine for transforming data items into a stream of data bits to

be sent to said client software component (col. 12, lines 20-27 'create a persistent object that is generally recorded to a file as a byte stream'), and a deserialization routine for transforming a stream of bits received from said client software components into data items (col. 12, lines 20-27 'file can be ... "rehydrated" back into its constituent objects').

Regarding Claim 24: The rejection of claim 21 is incorporated; further Allen discloses said first software component is installed in a first computer (Fig. 1, server 185), and wherein said one or more second software components are installed in one or more second computers (Fig. 1, client 100).

Claim Rejections - 35 USC § 103

15. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negated by the manner in which the invention was made.

16. **Claims 10-17, 25-27 and 31 are rejected under 35 U.S.C. 103(a) as being unpatentable over US 6,658,625 to Allen (Allen).**

Regarding Claim 10: Allen discloses a method for updating a software component in a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and a second software components (Fig. 1, Application 123), said method comprising:

providing said first software component with a data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD'); providing said first software component a dedicated second data exchange metadata description for any older version of said second software component that should be compatible (Fig. 1, Data Description 124); each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230'); said first software component serializing data items using data structures according to said first data exchange metadata description and said selected second data exchange metadata description (col. 11, lines 42-48 'parses and validates PCML data description 124 by using PCML DTD 230 ... produces an output object ... to call server program'); sending said serialized data items to said second software component (col. 14, lines 16-17 'causes the server program to run and produce an output that is sent to the client'), said first software component receiving serialized data items from said second software component and deserializing said data items (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired') using data structures according to said first data exchange metadata description and said selected second data exchange metadata description (col. 14, lines 12-13 'data converter ... would then use the data description'). Allen does not disclose said first software component identifying a version of a second software component but discloses a second software component identifying a version of

the first software component (col. 17, lines 33-36 'When data converter converts the data ... after reception from server A ... data definition 561 will be used for hostName'). Further Allen discloses his invention provides two-way communication between the first and second software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing between a plurality of second data exchange metadata descriptions, but teaches that metadata regarding a plurality of versions of said first software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562'). It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the first software component, because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

Regarding Claim 11: Allen discloses a method for exchanging data between software components in a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a server software component (Fig. 1, Server Program 195) and a client software component (Fig. 1, Application 123), said method comprising: providing said server software component with a first data exchange

metadata description containing information on data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD'); said server software component serializing data items on the basis of said first data exchange metadata description and said second data exchange metadata description (col. 11, lines 42-48 'parses and validates PCML data description 124 by using PCML DTD 230 ... produces an output object ... to call server program); sending said serialized data items to said client software component (col. 14, lines 16-17 'causes the server program to run and produce an output that is sent to the client'), said server software component receiving serialized data items from said client software component and deserializing said data items (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired') on the basis of said first data exchange metadata description and said second data exchange metadata description (col. 14, lines 12-13 'data converter ... would then use the data description').

Allen does not disclose said server software component identifying a version of a client software component but discloses a client software component identifying a version of the server software component (col. 17, lines 33-36 'When data converter converts the data ... after reception from server A ... data definition 561 will be used for hostName').

Further Allen discloses his invention provides two-way communication between the server and client software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing between a plurality of second data exchange metadata descriptions, but teaches that metadata regarding a plurality of versions of said server software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the server software component, because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

Regarding Claim 12: The rejection of claim 11 is incorporated; further, Allen discloses delivering said second data exchange metadata description during a data exchange session preformed between said server software component and said second software component (col. 12, lines 2-4 'This data is then converted ... when application 123 requests the data').

Regarding Claim 13: The rejection of claim 11 is incorporated; further, Allen discloses creating a serialization scheme (col. 12, lines 17-20 'serializing the hash table ... to create PCML serialized') for said server software component and said client software component on the basis of said metadata descriptions of these versions (col. 11, lines 42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ... PCML Serialized 220 is an enhancement to the parser output'); said serialization

scheme containing information needed in said server software component for the serialization of data to and the deserialization of data from said client software component (col. 11, line 66-col. 12, line 2 'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call server program 195 and to receive data ... from server program 195').

Regarding Claim 14: The rejection of claim 13 is incorporated; further, Allen discloses storing the created serialization scheme in said server software component (col. 13, lines 45-47 'the hash table ... be serialized ... and saved to a file'), using said stored serialization scheme in a subsequent data exchange session between said server software component and said identified version of said client software component (col. 13, lines 47-53 'the constructor will rehydrate the serialized hash table').

Regarding Claim 15: The rejection of claim 11 is incorporated; further Allen discloses said step of serializing comprises transforming data items into a stream of data bits to be sent to said client software component (col. 12, lines 20-27 'create a persistent object that is generally recorded to a file as a byte stream'), said step of deserializing comprises transforming a stream of bits received from said client software components into data items (col. 12, lines 20-27 'file can be ... "rehydrated" back into its constituent objects').

Regarding Claim 16: The rejection of claim 15 is incorporated; further, Allen discloses assigning a short identifier for each distinct instance of a data structure transferred in a data exchange operation (Fig. 4A-2, section 550 '<struct name="rwAccessList"'), sending and serializing that short identifier for each occurrence of each instance of a

data structure (col. 14, lines 21-25 'data description has already been parsed into a hash table ... and is essentially a "dictionary" of all the named elements'), serializing and sending each actual data structure instance only for the first occurrence within the data exchange operation instead of serializing the actual data structure instance for each occurrence within the data exchange operation (col. 14, lines 25-27 'a string object is passed to the hash table, which then returns a reference to the object represented by the string object').

Regarding Claim 17: Allen discloses a method for exchanging data between software components in a distributed software system comprising a first software component and a second software component; said method comprising storing a serialization scheme (col. 13, lines 45-47 'the hash table ... be serialized ... and saved to a file'), for at least one pair of said new version of said first software component and an older version of said second software component, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component (col. 11, line 66-col. 12, line 2 'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call server program 195 and to receive data ... from server program 195') and said serialization scheme being created on the basis of a first metadata description and a second metadata description (col. 11, lines 42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ... PCML Serialized 220 is an enhancement to the parser output') said first data exchange metadata description containing information on definitions of data structures to be used

in a serialized data exchange by said first software component (col. 11, lines 42-44 'PCML DTD') and said second data exchange metadata description of older version of said second software component containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230'); said first software component receiving serialized data items from said second software component and deserializing said data items on the basis of said serialization scheme (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired').

Allen does not disclose said first software component identifying a version of a second software component but discloses a second software component identifying a version of the first software component (col. 17, lines 33-36 'When data converter converts the data ... after reception from server A ... data definition 561 will be used for hostName'). Further Allen discloses his invention provides two-way communication between the first and second software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing serialization scheme corresponding to an identified version of said second software component, but teaches that metadata regarding a plurality of versions of said first software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and

choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the first software component, thereby creating multiple serialization schemes (col. 12, lines 17-20 'serializing the hash table'), because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

Regarding Claim 25: Allen discloses a server computer for a client-server computer system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising means for exchanging data with client software components (col. 3, lines 16-20 'data that is sent to and/or received from a server'), said means further comprising means for storing a first data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange by said server software (col. 11, lines 42-44 'PCML DTD'), said selected second data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said client software component (col. 11, lines 42-48 'parses and validates PCML data description 124 by using PCML DTD 230') means for serializing data items using data structures according to said first data exchange metadata description and said second data exchange metadata description (col. 11, lines 42-48 'parses and validates PCML data description 124 by using PCML DTD 230

Art Unit: 2124

... produces an output object ... to call server program), sending said serialized data items to said second software component (col. 14, lines 16-17 'causes the server program to run and produce an output that is sent to the client'), means for receiving serialized data items from said client software component and deserializing said data items (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired') on the basis of said first data exchange metadata description and said second data exchange metadata description (col. 14, lines 12-13 'data converter ... would then use the data description').

Allen does not disclose said first software component identifying a version of a second software component but discloses a second software component identifying a version of the first software component (col. 17, lines 33-36 'When data converter converts the data ... after reception from server A ... data definition 561 will be used for hostName'). Further Allen discloses his invention provides two-way communication between the first and second software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing between a plurality of second data exchange metadata descriptions, but teaches that metadata regarding a plurality of versions of said first software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562').

Art Unit: 2124

It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the first software component, because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

Regarding Claim 26: Allen discloses a computer for a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and at least one second software component (Fig. 1, Application 123), said computer comprising said first software component (Fig. 1, Server Program 195) and means for exchanging data with said at least one second software component (col. 3, lines 16-20 'data that is sent to and/or received from a server'), said means further comprising means for storing a serialization scheme (col. 13, lines 45-47 'the hash table ... be serialized ... and saved to a file'), for at least one pair of said new version of said first software component and a previous version of said second software component, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component (col. 11, line 66-col. 12, line 2 'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call server program 195 and to receive data ... from server program 195') and said serialization scheme being created on the basis of a first data exchange metadata description and a

Art Unit: 2124

second data exchange metadata description (col. 11, lines 42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ... PCML Serialized 220 is an enhancement to the parser output') said first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component (col. 11, lines 42-44 'PCML DTD') and said second data exchange metadata description of older version of said second software component containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230'); means for receiving serialized data items from said second software component and deserializing said data items on the basis of said selected serialization scheme (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired').

Allen does not disclose said first software component identifying a version of a second software component but discloses a second software component identifying a version of the first software component (col. 17, lines 33-36 'When data converter converts the data ... after reception from server A ... data definition 561 will be used for hostName'). Further Allen discloses his invention provides two-way communication between the first and second software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing serialization scheme corresponding to an identified version of said second software component, but teaches that metadata regarding a plurality of versions of said first software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the first software component, thereby creating multiple serialization schemes (col. 12, lines 17-20 'serializing the hash table'), because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

Regarding Claim 27: Allen discloses a data exchange metadata description containing information on data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD') serializing data items using data structures according to a second data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD') and said selected first data exchange metadata description (col. 11, lines 42-48 'parses and validates PCML data description 124 by using PCML DTD 230 ... produces an output object ... to call server program'); sending said serialized data items to said client software component (col. 14, lines 16-17 'causes the server program to run and

produce an output that is sent to the client'), receiving serialized data items from said client software component and deserializing said data items (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired') using data structures according to said first data exchange metadata description and said second data exchange metadata description (col. 14, lines 12-13 'data converter ... would then use the data description').

Allen does not disclose said first software component identifying a version of a second software component but discloses a second software component identifying a version of the first software component (col. 17, lines 33-36 'When data converter converts the data ... after reception from server A ... data definition 561 will be used for hostName'). Further Allen discloses his invention provides two-way communication between the first and second software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing between a plurality of second data exchange metadata descriptions, but teaches that metadata regarding a plurality of versions of said first software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the first

Art Unit: 2124

software component, because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

Regarding Claim 28: The rejection of claim 27 is incorporated; further, Allen discloses creating a serialization scheme (col. 12, lines 17-20 'serializing the hash table ... to create PCML serialized') for said server software component and said client software component on the basis of said metadata descriptions of these versions (col. 11, lines 42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ... PCML Serialized 220 is an enhancement to the parser output') said serialization scheme containing information needed in said server software component for the serialization of data to and the deserialization of data from said version of said second software component (col. 11, line 66-col. 12, line 2 'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call server program 195 and to receive data ... from server program 195')

Regarding Claim 27/29: The rejection of claim 27 is incorporated; further Allen discloses transforming data items into a stream of data bits to be sent to said client software component (col. 12, lines 20-27 'create a persistent object that is generally recorded to a file as a byte stream'), transforming a stream of bits received from said client software components into data items (col. 12, lines 20-27 'file can be ... "rehydrated" back into its constituent objects').

Regarding Claim 28/29: The rejection of claim 28 is incorporated; further Allen discloses transforming data items into a stream of data bits to be sent to said client

software component (col. 12, lines 20-27 'create a persistent object that is generally recorded to a file as a byte stream'), transforming a stream of bits received from said client software components into data items (col. 12, lines 20-27 'file can be ... "rehydrated" back into its constituent objects').

Regarding Claim 27/30: The rejection of claim 27 is incorporated; further Allen discloses assigning a short identifier for each distinct instance of a data structure transferred in a data exchange (Fig. 4A-2, section 550 '<struct name="rwAccessList"'), sending and serializing that short identifier of a data structure instead of the actual data structure when such short identifier is available (col. 14, lines 25-27 'a string object is passed to the hash table, which then returns a reference to the object represented by the string object').

Regarding Claim 28/30: The rejection of claim 28 is incorporated; further Allen discloses assigning a short identifier for each distinct instance of a data structure transferred in a data exchange (Fig. 4A-2, section 550 '<struct name="rwAccessList"'), sending and serializing that short identifier of a data structure instead of the actual data structure when such short identifier is available (col. 14, lines 25-27 'a string object is passed to the hash table, which then returns a reference to the object represented by the string object').

Regarding Claim 31: Allen discloses a server computer program comprising a program code configured to perform the following routines when run on a computer, storing a serialization scheme (col. 13, lines 45-47 'the hash table ... be serialized ... and saved to a file'), for at least one pair of said new version of said first software component and a

Art Unit: 2124

older version of said second software component, said serialization scheme containing information needed in said new version of said first software component for the serialization of data to and the deserialization of data from said respective previous version of said second software component (col. 11, line 66-col. 12, line 2

'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call server program 195 and to receive data ... from server program 195') and said serialization scheme being created on the basis of a first data exchange metadata description and a second data exchange metadata description (col. 11, lines 42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ... PCML Serialized 220 is an enhancement to the parser output') said first data exchange metadata description containing information on data structures to be used in a serialized data exchange by said first software component (col. 11, lines 42-44 'PCML DTD') and said second data exchange metadata description of older version of said second software component containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230'); receiving serialized data items from said second software component and deserializing said data items on the basis of said selected serialization scheme (col. 14, lines 21-25 'Some of the data elements that are returned by the server are converted here if desired').

Allen does not disclose said first software component identifying a version of a second software component but discloses a second software component identifying a version of the first software component (col. 17, lines 33-36 'When data converter converts the

Art Unit: 2124

data ... after reception from server A ... data definition 561 will be used for hostName').

Further Allen discloses his invention provides two-way communication between the first and second software components (col. 8, lines 29-30 'input and output data sent to and received from server program 195') thereby making it inherent that the process disclosed is reversible.

Further, Allen does not disclose choosing serialization scheme corresponding to an identified version of said second software component, but teaches that metadata regarding a plurality of versions of said first software component is contained within a single data exchange metadata description (Fig. 4A-2 data definitions 561 and 562) and choosing between these entries (col. 17, lines 33-42 'data definition 561 will be used ... instead of data definition 562').

It would have been obvious to a person of ordinary skill in the art at the time of the invention to split the data exchange metadata description disclosed in Allen (Fig. 4A) into multiple files each containing only data regarding a single version of the first software component, thereby creating multiple serialization schemes (col. 12, lines 17-20 'serializing the hash table'), because one of ordinary skill in the art would have been motivated to ease maintenance by maintaining separate DTDs for different server versions (col. 4, lines 19-22 'broken down into a set of autonomous entities').

17. Claims 3 and 6-9 are rejected under 35 U.S.C. 103(a) as being unpatentable over US 6,658,625 to Allen (Allen) in view of 2001/0,049,743 to Phippen et al. (Phippen).

Regarding Claim 1/3: The rejection of claim 1 is incorporated; further Allen does not disclose verifying the backward compatibility of said new version of said first software component, but does disclose different versions of software requiring different formats of data (col. 17, lines 21-23 'indicates that hostname has a fixed length of 20 bytes for all releases ... of version one').

Phippen teaches verifying compatibility of a message being sent from a first software component to a second software component (par. [0008] 'means for determining compatibility of ... input message formats with ... output message formats') based on associated metadata descriptions (par. [0014] 'Compatibility is most easily determined from meta-data'), in an analogous art for the purpose of 'providing a message transformation selection tool' (par. [0007]).

It would have been obvious to a person of ordinary skill in the art at the time of the invention to apply the 'means for determining compatibility' disclosed in Phippen (par. [0008]) to the messages being sent between Allen's server (Fig. 1, server application 185) and client (Fig. 1, application 123) applications because one of ordinary skill in the art would have been motivated to ensure compatibility (par. [0008] 'means for determining compatibility').

Regarding Claim 2/3: The rejection of claim 1 is incorporated; further Allen does not disclose verifying the backward compatibility of said new version of said first software component, but does disclose different versions of software requiring different formats of data (col. 17, lines 21-23 'indicates that hostname has a fixed length of 20 bytes for all releases ... of version one').

Art Unit: 2124

Phippen teaches verifying compatibility of a message being sent from a first software component to a second software component (par. [0008] 'means for determining compatibility of ... input message formats with ... output message formats') based on associated metadata descriptions (par. [0014] 'Compatibility is most easily determined from meta-data'), in an analogous art for the purpose of 'providing a message transformation selection tool' (par. [0007]).

It would have been obvious to a person of ordinary skill in the art at the time of the invention to apply the 'means for determining compatibility' disclosed in Phippen (par. [0008]) to the messages being sent between Allen's server (Fig. 1, server application 185) and client (Fig. 1, application 123) applications because one of ordinary skill in the art would have been motivated to ensure compatibility (par. [0008] 'means for determining compatibility').

Regarding Claim 6: The rejection of claim 5 is incorporated; further, Allen does not disclose verifying the backward compatibility of said new version of said first software component, but does disclose different versions of software requiring different formats of data (col. 17, lines 21-23 'indicates that hostname has a fixed length of 20 bytes for all releases ... of version one').

Phippen teaches verifying compatibility of a message being sent from a first software component to a second software component (par. [0008] 'means for determining compatibility of ... input message formats with ... output message formats') based on associated metadata descriptions (par. [0014] 'Compatibility is most easily determined

from meta-data'), in an analogous art for the purpose of 'providing a message transformation selection tool' (par. [0007]).

It would have been obvious to a person of ordinary skill in the art at the time of the invention to apply the 'means for determining compatibility' disclosed in Phippen (par. [0008]) to the messages being sent between Allen's server (Fig. 1, server application 185) and client (Fig. 1, application 123) applications because one of ordinary skill in the art would have been motivated to ensure compatibility (par. [0008] 'means for determining compatibility').

Regarding Claim 7: Allen discloses a method for updating a software component in a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and one or more second software components (Fig. 1, Application 123), installed apart from each other (Fig. 1, Network I/F 163) said method comprising: updating said first software component with a new version of software (col. 6, lines 42-44 'multiple versions of server programs') substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components (col. 3, lines 7-10 'interprets a data description ... that can accommodate changes in the data'); providing said updated software version with a data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD'); providing a dedicated data exchange metadata description for each version of said second software

component that should be compatible (Fig. 1, Data Description 124); each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230').

Allen does not disclose verifying the backward compatibility of said new version of said first software component, but does disclose different versions of software requiring different formats of data (col. 17, lines 21-23 'indicates that hostname has a fixed length of 20 bytes for all releases ... of version one').

Phippen teaches verifying compatibility of a message being sent from a first software component to a second software component (par. [0008] 'means for determining compatibility of ... input message formats with ... output message formats') based on associated metadata descriptions (par. [0014] 'Compatibility is most easily determined from meta-data'), in an analogous art for the purpose of 'providing a message transformation selection tool' (par. [0007]).

It would have been obvious to a person of ordinary skill in the art at the time of the invention to apply the 'means for determining compatibility' disclosed in Phippen (par. [0008]) to the messages being sent between Allen's server (Fig. 1, server application 185) and client (Fig. 1, application 123) applications because one of ordinary skill in the art would have been motivated to ensure compatibility (par. [0008] 'means for determining compatibility').

Regarding Claim 8: Allen discloses a method for updating a software component in a distributed software system (col. 7, lines 14-16 'particularly useful in a client-server architecture') comprising a first software component (Fig. 1, Server Program 195) and one or more second software components (Fig. 1, Application 123), installed apart from each other (Fig. 1, Network I/F 163) said method comprising: updating said first software component with a new version of software (col. 6, lines 42-44 'multiple versions of server programs') substantially compatible with the functionality of a non-updated version(s) of said one or more second software components but having data structures incompatible with said non-updated version(s) of said one or more second software components (col. 3, lines 7-10 'interprets a data description ... that can accommodate changes in the data'); providing said updated software version with a data exchange metadata description containing information on definitions of data structures to be used in a serialized data exchange (col. 11, lines 42-44 'PCML DTD'); providing a dedicated data exchange metadata description for each version of said second software component that should be compatible (Fig. 1, Data Description 124) each said data exchange metadata description containing information on data structures to be used in a serialized data exchange by the respective version of said second software component (col. 11, lines 42-48 parses and validates PCML data description 124 by using PCML DTD 230'); creating a serialization scheme (col. 12, lines 17-20 'serializing the hash table ... to create PCML serialized') for at least one pair of said new version of said first software component and a previous version of said second software component on the basis of said metadata descriptions of these versions (col. 11, lines

Art Unit: 2124

42-45 'parses and validates PCML data description 124 by using PCML DTD 230 ...

PCML Serialized 220 is an enhancement to the parser output'); said serialization

scheme containing information needed in said new version of said first software

component for the serialization of data to and the deserialization of data from said

respective previous version of said second software component (col. 11, line 66-col. 12,

line 2 'ProgramCallDocument class 128 uses ... the output ... of XML parser 125 to call

server program 195 and to receive data ... from server program 195'); delivering said at

least one serialization scheme with said new version of said software component (col.

12, lines 28-44 'a software engineer who is writing a PCML data description ... would

call the XML Parser class ... would then be serialized').

Allen does not disclose verifying the backward compatibility of said new version of said

first software component, but does disclose different versions of software requiring

different formats of data (col. 17, lines 21-23 'indicates that hostname has a fixed length

of 20 bytes for all releases ... of version one').

Phippen teaches verifying compatibility of a message being sent from a first software

component to a second software component (par. [0008] 'means for determining

compatibility of ... input message formats with ... output message formats') based on

associated metadata descriptions (par. [0014] 'Compatibility is most easily determined

from meta-data'), in an analogous art for the purpose of 'providing a message

transformation selection tool' (par. [0007]).

It would have been obvious to a person of ordinary skill in the art at the time of the

invention to apply the 'means for determining compatibility' disclosed in Phippen (par.

Art Unit: 2124

[0008]) to the messages being sent between Allen's server (Fig. 1, server application 185) and client (Fig. 1, application 123) applications because one of ordinary skill in the art would have been motivated to ensure compatibility (par. [0008] 'means for determining compatibility').

Regarding Claim 9: The rejection of claim 8 is incorporated; further, Allen discloses default values for data items (col. 16, line 35 'The input data can have initial values').

Allen does not disclose verifying the backward compatibility of said new version of said first software component but does disclose different versions of software requiring different formats of data (col. 17, lines 21-23 'indicates that hostname has a fixed length of 20 bytes for all releases ... of version one').

Phippen teaches comparing the identifier and type information of data structures in said new version and said previous version (pars. [0047]-[0049] 'if the type of Fa is compatible with the type of Fb'), in an analogous art for the purpose of 'providing a message transformation selection tool' (par. [0007]).

It would have been obvious to a person of ordinary skill in the art at the time of the invention to apply the 'means for determining compatibility' disclosed in Phippen (pars. [0047]-[0049]) to the messages being sent between Allen's server (Fig. 1, server application 185) and client (Fig. 1, application 123) applications because one of ordinary skill in the art would have been motivated to ensure compatibility (par. [0008] 'means for determining compatibility').


Conclusion

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. US 5,956,688 to Kokubo et al.; US 5,999,938 to Bliss et al.; US 6,249,794 to Raman; US 6,662,186 to Esquibel; US 2001/0047385 to Tuatini; 2003/0028540 to Lindberg et al.; and US 2003/0028447 to O'Brien et al.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Jason Mitchell whose telephone number is (571) 272-3728. The examiner can normally be reached on Monday-Thursday and alternate Fridays 7:30-5:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Kakali Chaki can be reached on (571) 272-3719. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306. Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see <http://pair-direct.uspto.gov>. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Jason Mitchell
12/22/04



**TODD INGBERG
PRIMARY EXAMINER**